REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED

1b. RESTRICTIVE MARKINGS:

2a. SECURITY CLASSIFICATION AUTHORITY:

2b. DECLASSIFICATION/DOWNGRADING SCHEDULE:

3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution unlimited.

4. PERFORMING ORGANIZATION REPORT NUMBER(S): AD-A212 848

5. MONITORING ORGANIZATION REPORT NUMBER(S): AFOSR-TR- 89-1254

6a. NAME OF PERFORMING ORGANIZATION: University of Missouri

6b. OFFICE SYMBOL (If applicable):

7a. NAME OF MONITORING ORGANIZATION: Air Force Office of Scientific Research

6c. ADDRESS (City, State, and ZIP Code): Dept. of Math and Computer Science, Columbia, MO 21228

7b. ADDRESS (City, State, and ZIP Code): Building 410, Bolling AFB, DC 20332-6448

8a. NAME OF FUNDING/SPONSORING ORGANIZATION: AFOSR

8b. OFFICE SYMBOL (If applicable): NM

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: AFOSR-86-0124

8c. ADDRESS (City, State, and ZIP Code): Building 410, Bolling AFB, DC 20332-6448

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.
61102F	2304	A8	

11. TITLE (Include Security Classification): Iterative Methods For Linear Complementary and Related Problems

12. PERSONAL AUTHOR(S): Dr Tzong Huei Shiau

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM 15 May 86 TO 14 May 89		

16. SUPPLEMENTARY NOTATION:

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number):

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

A sparsity preserving LP-based SOR method for solving classes of linear complementary problems including the case where the given matrix is positive semidefinite is proposed. The LP subproblems need be solved only approximately by a SOR method. Heuristic enhancement is discussed. Numerical results for a special class of problems are presented, which show that the heuristic enhancement is very effective and the resulting program can solve problems of more than 100 variables in a few seconds even on a personal computer. Keywords: Linear Programming; Successive Overrelaxation Method. (jhd)

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: ☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL: Dr. Neal D. Glassman

22b. TELEPHONE (Include Area Code): (202) 767-5026

22c. OFFICE SYMBOL: NM

DD Form 1473, JUN 86 — Previous editions are obsolete. — SECURITY CLASSIFICATION OF THIS PAGE

To appear in Journal of Optimization Theory and Applications.

An LP-Based Successive Overrelaxation Method for Linear Complementarity Problems[1,2]

T. H. SHIAU[3]

Communicated by O. L. Mangasarian

ITERATIVE METHODS FOR LINEAR COMPLEMENTARY AND RELATED PROBLEMS

---

[1]This research was sponsored by the Air Force Office of Scientific Research, Grant No. AFOSR-86-0124. Part of this material is based on work supported by the National Science Foundation under Grant No. MCS-82-00632.

[2]The author is grateful to Dr. R. De Leone for his helpful and constructive comments on this paper.

[3]Assistant Professor, Department of Computer Science, University of Missouri-Columbia, Columbia, Missouri

Abstract. A sparsity preserving LP-based SOR method for solving classes of linear complementarity problems including the case where the given matrix is positive semidefinite is proposed. The LP subproblems need be solved only approximately by a SOR method. Heuristic enhancement is discussed. Numerical results for a special class of problems are presented, which show that the heuristic enhancement is very effective and the resulting program can solve problems of more than 100 variables in a few seconds even on a personal computer.

Key Words. Complementarity problems, iterative methods, quadratic programming, successive overrelaxation methods.

1. Introduction

We consider iterative methods for solving the linear complementarity problem (LCP):

$$\text{Find } x \text{ in } R^n : x \geq 0,\ Mx+q \geq 0,\ x^T(Mx+q) = 0 \qquad (1)$$

where M is a given n-by-n real matrix and q is a given real n-vector. The general LCP is NP-complete (Ref. 1) and therefore very difficult to solve even for problems with moderate size, say n = 50. However, in many applications the matrix M has some nice properties, e.g. M is positive semidefinite or all its principal minors are positive (i.e. M is a P-matrix), and the problem can be solved by some direct pivoting methods, e.g. the principal pivoting method (Ref. 2) or Lemke's method (Ref. 3), or a semi-iterative pivoting method by Shiau (Ref. 4).

Although a pivoting method gives a very accurate solution in a finite number of steps, it may not be suitable for some large sparse problems. Since pivoting can easily destroy the sparsity after a number of steps, $O(n^2)$ storage space is needed for the pivoting even if M has only O(n) nonzero entries. Iterative methods that preserve sparsity and work only on the nonzero entries of the input data are much more attractive for very large problems, for the storage requirement is greatly reduced and the processing could be faster.

Many iterative methods have been proposed (e.g. Refs. 5-9). However, they were developed primarily for the case that M has some kind of strict positivities. For example, Mangasarian's algorithm (Ref. 9) converges when M is symmetric and strictly co-positive, or copositive-plus and there exists an x such that $MX+q>0$. In one of the most important applications where the LCP is formulated for solving the underlying quadratic program (Ref. 2), the matrix M is only positive semidefinite and not definite. (There are special cases, e.g. the quadratic program is strictly convex and separable, that existing methods may apply (e.g. Refs. 10-12)). In this work, we present a sparsity-preserving iterative method that can solve nonsymmetric LCP's when M is only positive semi-definite.

We give a brief word about notation. The n-dimensional Euclidean space is denoted by R^n. Vectors in R^n are column vectors and denoted by x, y, z, etc. Superscripts are used to denote different vectors such as x^0, x^1, but the superscript T denotes the transpose, e.g. x^T, $(x^k)^T$, M^T.

2. Method and Convergence Theorem

Let us consider first the following quadratic program:

$$0 = \min f(x) \quad \text{subject to } x \in S, \tag{2}$$

where f is a convex quadratic function, S is defined by a finite number of linear inequalities, and $0 = f(\bar{x}) \leq f(x)$ for some $\bar{x}$ in S, for all x in S. The intension is to set $S = \{x | x \geq 0, Mx+q \geq 0\}$ and $f(x) = x^T(Mx+q)$, then all the assumptions of f and S are satisfied provided that M is positive semidefinite and the LCP has a solution. In this setting, every solution of (2) is a solution of (1) and vice versa. We present the following iterative scheme and its convergence theorem. The proof of a more general scheme can be found in (Ref. 4) and is very similar to that In (Ref. 13). We include the proof here to make the paper self-contained.

2.1 Iterative Scheme

Step 1. Given $x^k \in S$, stop if $f(x^k) = 0$ (within a fixed tolerance). Otherwise, find y^k satisfying

$$y \in S, \quad f(x^k) + \nabla f(x^k)^T(y^k - x^k) \geq 0 \tag{3}$$

Step 2. Let $p^k = y^k - x^k$ be the direction of search, find $x^{k+1} = x^k + t_k p^k$ where t_k is defined by

$$t_k = \underset{0 \leq t \leq 1}{\arg \text{minimize}} \; f(x^k + tp^k) \tag{4}$$

Step 3. Go to Step 1.

The computation of y^k is discussed in the next section. Since $f(x)$ is quadratic, the objective function of (4) is a quadratic function of t, in fact,

$$f(x^k + tp^k) = f(x^k) + (\nabla f(x^k)^T p^k)t + \tfrac{1}{2} t^2 (p^k)^T \nabla^2 f(x^k) p^k$$

$$=: a_k + b_k t + \tfrac{1}{2} c_k t^2 \qquad (5)$$

Note that $b \le -f(x) < 0$ by (3), and that $c_k \ge 0$ by the convexity of $f(x)$. It is very easy to compute t_k as shown by the following lemma.

Lemma 2.1. Let $g(t) = a + bt + \tfrac{1}{2} ct^2$, where $b<o$, $c \ge o$. Then

$$\bar{t} := \begin{cases} -b/c, & \text{if } c > -b, \\ 1, & \text{otherwise,} \end{cases}$$ is the minimizer of the following problem.

minimize $g(t)$, (6a)

$0 \le t \le 1$ (6b)

moreover,

$$g(0) - g(\bar{t}) \ge -\tfrac{1}{2} b \bar{t}. \qquad (7)$$

Proof the case $c = o$ is trival. Suppose $c>0$.

So $g'(t)<0$ for $0 \le t < -b/c$, and $g'(t)>0$ for $t>-b/c$. Therefore $g(t)$ is strictly decreasing on $[0,-b/c]$ and increasing on $[-b/c,+\infty]$. If $-b/c < 1$, i.e. $c>-b$, then $\bar{t} = -b/c$ solves (6) and it is straightforward to see that (7) holds as an equality. If $-b/c \ge 1$, then $\bar{t} = 1$ is the constrained minimum and

$$g(0)-g(1) = -b-\tfrac{1}{2} c \ge -b/2$$

proving (7).

Theorem 2.1. Convergence Theorem. Assume that $f(x)$ is convex and quadratic, S is nonempty and $0 = \min_{x \in S} f(x)$. Let $\{x^k, y^k\}$ be generated as in 2.1. If $\{y^k\}$ is bounded, then $\{f(x^k)\}$ converges monotonously to zero.

Proof By Lemma 2.1,

$$f(x^k) - f(x^{k+1}) \geq -\tfrac{1}{2} t_k b_k \geq \tfrac{1}{2} t_k f(x^k) \tag{8}$$

where b_k is defined in (5), and the last inequality follows by (3). Summing up (8) for $k = 0, 1, \ldots, N$, we have

$$f(x^0) - f(x^{N+1}) \geq \tfrac{1}{2} \sum_{k=0}^{N} t_k f(x^k)$$

Since the left-hand side is bounded for all N, the positive series $\sum_{k=0}^{\infty} t_k f(x^k)$ converges. Hence

$$\lim_{k \to \infty} t_k f(x^k) = 0 \tag{9}$$

By Lemma 2.1, t_k is either 1 or $t_k = -b_k/c_k$. By assumption $\{y^k\}$ is bounded, so is $\{x^k\}$ since x^k is a convex combination of x^0 and y^j, $j = 0, 1, \ldots, k-1$. Therefore, $\{c_k\}$ is also bounded.

So

$$t_k \geq \min \{1, b_k/V\} \geq \min \{1, f(x^k)/V\} \tag{10}$$

where V is a bound of $\{c_k\}$. Since $\{f(x^k)\}$ is monotonously decreasing by (8), if it did not converge to zero, then $f(x^k) \geq \delta > 0$ for all k, for some δ. Then, by (10), $t_k f(x^k) \geq \min \{1, f(x^k)/V\} \cdot f(x^k) \geq \min \{\delta, \delta^2/V\}$,

which contradicts to (9). □

Corollary 2.1. If we set $f(x) = x^T(Mx+q)$, $S = \{x | x \geq 0, Mx+q \geq 0\}$, where M is positive semidefinite. Assume S is nonempty and $\{y^k\}$ is bounded, then $\lim_{k\to\infty} d(x^k, \bar{S}) = 0$

where $\bar{S}$ is the solution set of (1), and $d(x,\bar{S})$ is the distance between x and $\bar{S}$ defined by

$$d(x,\bar{S}) = \inf_{y\in\bar{S}} ||x-y||$$

Proof It follows by Theorem 2.1 and Corollary 2.1 of (Ref. 14), which shows that for $x \in S$

$$d(x,\bar{S}) \leq O(f(x) + \sqrt{f(x)}).$$ □

The inequality in (3) can be considered as a cutting plane which cuts off part of S including the current point x^k, and the remaining part still contains the whole $\bar{S}$, because of the convexity of f(x). In the case that M is not positive semidefinite, but has other positivity properties such as that all principal

minors of which are positive, the scheme can be modified by a weaker cutting plane which cuts off less than the one in (3) does, so as to guarantee that y^k exists, The new cut is defined by

$$\theta_k f(x^k) + \nabla f(x^k)^T(x-x^k) \leq 0$$

where $\theta_k \leq 1$. The number θ_k should be as large as is allowed (for the existence of y^k) to speed up the algorithm. For simplicity of the paper, we omit this case and concentrate on that M is positive semidefinite, thereby θ_k is always chosen to be 1.

3. Solving y^k By SOR

The main effort in the iterative scheme of 2.1 for solving (1) is the computation of y^k satisfying (3). This is to find a feasible point, in the k-th iteration, of the system of linear constraints:

$$My + q \geq 0,\ y \geq 0,\ \nabla f(x^k)^T(y-x^k) \leq -f(x^k) \qquad (11)$$

Note that x^k and y^{k-1} satisfy all but the last inequality, i.e. the cut, and therefore they can be selected as starting points for solving y^k. There are a few non-pivoting methods for solving (11) which involve the inversion of a matrix in each iteration. For example, by replacing the last inequality by equality, one can easily employ the basic (single phase, no sliding variable) Karmarkar method (Ref. 15) to:

$$\nabla f(x^k)^T x^k - f(x^k) = \min \nabla f(x^k)^T y$$
$$\text{s.t. } My + q \geq 0,\ y \geq 0$$

We propose to use a successive overrelaxation method (SOR) which needs no matrix inversion and is sparsity-preserving (Refs. 16-18). The name SOR is more commonly referred to the relaxation method for solving systems of n linear equations in n unknowns (e.g. see Refs. 19-20). The SOR for system (11) of linear inequalities is different but similar.

3.1 SOR for Systems of Linear Inequalities

Problem 3.1: Find y in R^n satisfying $A_i^T Y \leq b$, $i = 1, 2, \ldots, m$, where $A_i \in R^n$, $b_i \in R$ are given.

Algorithm 3.1 Step 1. Given z^j, $j \geq 0$, stop if z^j satisfies (within a tolerance) all the inequalities. Otherwise, pick the next inequality which z^j violates, say $A_i^T Y \leq b_i$, compute

$$z^{j+1} := z^j + \mu((b_i - A_i^T z^j)/A_i^T A_i)A_i \qquad (12)$$

where $0 < \mu \leq 2$.

Step 2. Go to Step 1.

In Step 1, the selection of the next violated inequality is based on the cyclic order 1, 2,...,m, 1, 2,... One alternative is to choose the one that z violates most, i.e. choose i so that $(b_i - A_i^T z^j)/\| A_i \|$ is most negative. The strict cyclic order can be relaxed to almost cyclic order defined as follows. Let $i_1, i_2, \ldots, i_\ell, \ldots$ be the indices of the inequalities to be checked for possible violation and relaxation (12), the order is called almost cyclic if $1 \leq i_\ell \leq m$, and

there exists an integer C such that $\{1,2,\ldots,m\} \subseteq \{i_\ell, i_{\ell+1} \ldots, i_{\ell+C}\}$ for all $\ell \geq 1$. The strict cyclic order is almost cyclic with C = m (Ref. 18). When $\mu = 1$, (12) moves z^j along $-A_i$ to z^{j+1} to satisfy the i-th constraint. The constraint is said to be relaxed in the process. It is an overrelaxation if $\mu > 1$, and an underrelaxation if $\mu < 1$.

Theorem 3.1

Let $\bar{y}$ be any solution of the Problem 3.1, let $\{z^j\}$ be generated by the SOR method (with an almost cyclic order or most-violation rule).

(a) If $0 < \mu < 2$, then $|z^{j+1} - \bar{y}| < |z^j - \bar{y}|$, and $\{z^j\}$ converges to a solution.

(b) If there exists y such that $A_i^T Y < b_i$ for $i = 1,2,\ldots,m$, then by choosing $\mu = 2$, the algorithm will find a solution in a finite number of iterations.

Proof See (Refs. 16-18). □

One can vary μ in each iteration, say, choose μ_j for the j-th iteration. The conclusion of (a) remains true if $0 < \varepsilon \leq \mu_j \leq 2 - \varepsilon$ for all j for some ε.

4. Computational Details and Heuristic Enhancements

4.1 Since SOR is an infinite procedure, we may not solve y^k exactly. However, we can get a point feasible to within a small tolerance in a finite number of iterations. Since any solutions

of the LCP (1) satisfies (11), each SOR iteration moves the point closer to the solution set by Theorem 3.1(a). The overall solution process is, therefore, applying the SOR iteration (12) until the movement is small, followed by the line-search to find x and updating the cutting plane, then resuming the SOR iteration. If we select y^k instead of x^{k+1} as the starting point when SOR restarts, then we have the monotonously decreasing of $\{d(z^j, \bar{S})\}$ as well as $\{f(x^k)\}$, where $\{z^j\}$ refers to all the points generated by SOR which contains $\{y^k\}$ as a subsequence.

4.2 When applying SOR, the n nonnegativity constraints $x \geq 0$ can be relaxed easily by replacing the negative components of z^j with zeroes or small positive numbers, for overrelaxation. Hence, they can be relaxed as often as possible, i.e. every time any component of z^j becomes negative after relaxing one of the rest n+1 constraints, that component can be made nonnegative. The convergence of $\{z^j\}$ still holds, for the resulting order is almost cyclic (defined in 3.1 with $C = n(n+1)$).

4.3 Heuristically, if a constraint $x_i \geq 0$ is violated very often, we can set $x_i = 0$, thereby reduce the problem size. The reasoning is that if $\{z^j\}$ converges to $\bar{x}$ and $\bar{x}_i > 0$ were nonzero, then z_i^j should also be positive for all but finitely many j. Similarly, if $w_i \geq 0$, where $(w_1, w_2, \ldots, w_n)^T := Mx+q$, is violated very often, we can set $w_i = 0$, i.e. solve $(Mx+q)_i = 0$ for some x_ℓ and thereby eliminate variable x_ℓ and constraints $x_\ell \geq 0$, $w_i \geq 0$. This clearly requires some computation, but it can easily be justified if M is very sparse. Since we also relax the nonnegativity constraints of x_ℓ, x_ℓ should

be chosen among only variables for which the current values are sufficiently large. It is possible that we may erroneously set x_i or w_i to zero, but we can easily back up by restoring the variables and constraints. For example, if the decreasing of $f(x^k)$ becomes unsatisfactorily slow after setting $w_i = 0$ and removing x_ℓ as a consequence, we can compute the value of x_ℓ by the equation $w_i = 0$, restore the constraints $w_i \geq 0$, $x_\ell \geq 0$, and resume the computation of x_ℓ in the SOR iteration.

By similar arguments if a constraint $x_i \geq 0$ (or $w_i \geq 0$) has not been violated lately, we may set $w_i = 0$ (or $x_i = 0$).

4.4 The line-search computes x^{k+1} as the minimizer of $f(x)$ on the line segment with endpoints x^k and y^k. We may do higher dimensional searches, e.g. by minimizing $f(x)$ on the triangle with vertices x^k, y^{k-1}, and y^k. This is not much harder than the line-search since $f(x)$ is quadratic.

4.5 Strictly speaking, y^k does not satisfy (11) exactly, rather, it satisfies:

$$y^k \in S_k := \{y \mid My + q + \varepsilon_k e \geq 0,\ y + \varepsilon_k e \geq 0\}, \nabla f(x^k)^T (y^k - x^k) \leq -f(x^k)$$

where $\varepsilon_k > 0$ is the tolerance in iteration k, and e is the vector of ones. However, after replacing S in (3) by S_k, Convergence Theorem 2.1 remains valid, i.e., we still have $f(x^k) \downarrow 0$. Corollary 2.1 needs to be modified. If we choose ε_k so that $\varepsilon_k \to 0$, we can still have $\lim_{k\to\infty} d(x^k, \bar{S}) = 0$, by Theorem 2.7 of (Ref. 14).

5. Numerical Results

A computer program is written on an IBM personal computer (PC AT) for a special class of LCPs for which the matrix is of the form:

$$M = \begin{bmatrix} d_1 & & & \\ & d_2 & & A^T \\ & & \ddots & \\ & & & d_N \\ \hline -A & & & 0 \end{bmatrix}$$

where d_i is either 1 or 0, and A is an N-by-E node-arc incidence matrix in which there are exactly two nonzero entries per column, a 1 and a -1. Any convex separable quadratic network flow problem with N nodes and E edges can be solved by such an LCP. The major reason we choose this LCP in our numerical study is that when the heuristic approach discussed in the previous section decides to set some x_i or w_i to zero, it is very easy to do so to reduce the problem size. For more general problems, it may need more computational effort and storage space in doing that. Beyond that, we see no significant difference for the iterative method and its heuristics in solving LCPs with general sparse positive semidefinite matrices. So the following results should reflect the performance of the method not only for the special problem, but also for the general problems.

Many problems have been tested, we include here output of five runs of different problem size ranges and different methods for choosing the tolerance in solving (11). The following remarks explain what each column means. See Table 1-5.

Remark 5.1. (Column 1) All problems are solved in less than seven iterations.

Remark 5.2. (Column 2) As stated in Section 2.1, the stopping criterion is that $|f(x)|$ be smaller than a fixed tolerance, called Initial Tolerance in each output, for which we use $||q||_1/1000n$ and find it satisfactory. The initial tolerance is also used in solving (11) for the first y. In all except Table 2 for which the same tolerance is used in each iteration in solving (11), the tolerance is halved after each iteration. One extra iteration is done to bring the infeasibility within "user specified" tolerance, called Final Tolerance in the outputs.

Remark 5.3. (Column 3) The program is given the solution so that it can compute the square of the relative error $||x^k-x^*||_2^2/||x^*||_2^2$ We choose to avoid computing the square root. Note that the solution procedure is in no way dependent on the given solution. In fact, the time for computing this column should not have been counted.

Remark 5.4. (Column 4) In solving (11), a cycle is complete when all of the n constraints $w_i \geq 0$ and the cut have been checked and relaxed if violation is found.

Remark 5.5. (Column 5) This is the total number of changes of y in solving (11). Therefore it is also the total number of times of all constraints violated.

Remark 5.6. (Column 6) This is the clock that starts at 00:00.00 in each run. The first two digits are in minutes, the last two are in hundreds of seconds.

Remark 5.7. Table 2 tells us that high accuracy cannot be achieved without reducing the tolerance, which is consistent with theoretical results of (Ref. 14). But when the tolerance is reduced iteratively in Table 3, the computing time is increased greatly, so is the number in Column 5. The latter means that many constraints are violated many times, but then many w_i and x_i will be set to zero by the heuristic enhancement criteria. The problem size can therefore be reduced rapidly, that is reflected in Table 4 in which the computing time is less than one percent of that in Table 3. The same remarkable performance is found in Table 5, in which a problem of the total of 116 variables is solved with relative accuracy of 10^{-7} in just 32 seconds--by a personal computer.

6. Additional Observations and Summary

In all our test problems of size more than 50, starting at the second iteration all constraints violated more than 5 times turned out to be active, and more than 90% of the active constraints are violated more than 50 times in a single iteration. In other words, the program has picked, without a single error, more than 90% of the active constraints in just three iterations. Assuming

the same performance, the method would solve problems of millions of variables in minutes or seconds on mainframes or supercomputers. The related problem of how to reduce the problem size for general LCP while keeping as much sparsity as possible, when many of the active constraints are known, becomes a very interesting problem which we leave to the future.

Table 1. Output By LCPSOR, no heuristic enhancement.

Iteration	f(x)	Square of rel. error	Number of cycles	No. of SOR iterations	Clock
0	67.136070	2.808698E-01	3	13	00:00.05
1	10.586370	1.373864E-02	5	23	00:00.16
2	0.608786	8.386110E-05	17	112	00:00.49
3	-0.000018	2.790467E-09	28	190	00:01.04
4	-0.000044	1.886973E-13	53	321	00:02.09

Number of nodes = 3; Number of edges = 4.

Initial tolerance = 0.004000; Final tolerance = 0.000005 .

Table 2. Fixed tolerance, no heuristic enhancement.

Iteration	f(x)	Square of rel. error	Number of cycles	No. of SOR iterations	Clock
0	2369.166000	1.815431E-01	4	46	00:00.33
1	348.715500	1.577661E-02	6	92	00:01.10
2	19.965990	2.935577E-03	63	848	00:07.36
3	3.917515	7.512022E-04	314	4235	00:37.79
4	0.291592	1.806640E-04	747	7048	01:46.11
5	0.108357	1.318387E-04	280	1678	02:10.45
6	-0.082139	1.292340E-04	25	129	02:12.75
7	-0.082139	1.292340E-04	0	0	02:12.92

Number of nodes = 20; Number of edges = 38.

Fixed tolerance = 0.012897 .

Table 3. Reducing tolerance, no heuristic enhancement.

Iteration	f(x)	Square of rel. error	Number of cycles	No. of SOR iterations	Clock
0	2369.166000	1.815431E-01	4	46	00:00.33
1	349.100200	1.576520E-02	6	94	00:01.10
2	20.196930	2.908019E-03	74	1211	00:08.73
3	4.116029	7.416783E-04	513	8657	01:00.91
4	0.738284	7.504172E-05	1807	32150	04:06.95
5	0.106307	2.941542E-06	4287	69869	11:19.48
6	0.004395	8.453883E-08	5070	70660	19:25.02
7	0.000147	7.029873E-11	10185	195481	37:23.54

Same problem as in Table 2.

Initial tolerance = 0.012897; Final tolerance = 0.000005 .

Table 4. Output By LCPSOR with heuristic enhancement.

Iteration	f(x)	Square of rel. error	Number of cycles	No. of SOR iterations	Clock
0	2369.166000	1.815431E-01	4	46	00:00.44
1	349.100200	1.576520E-02	6	94	00:01.43
2	20.196930	2.908019E-03	74	1211	00:10.77
3	3.963225	4.940340E-04	18	82	00:11.26
4	0.338470	2.348035E-06	39	216	00:12.25
5	-0.002658	8.072329E-11	93	434	00:13.73
6	-0.000023	1.349873E-14	98	422	00:14.21

Same problem, same initial and final tolerance as in Table 3.

Table 5. Output By LCPSOR with heuristic enhancement.

Iteration	f(x)	Square of rel. error	Number of cycles	No. of SOR iterations	Clock
0	3990.690000	1.634287E-01	8	113	00:01.32
1	627.943400	1.455267E-02	6	159	00:03.19
2	40.228890	3.347591E-03	71	2540	00:21.31
3	7.471679	2.231230E-04	14	130	00:21.97
4	0.767882	2.733713E-06	36	436	00:23.51
5	0.010428	5.249066E-09	123	1146	00:26.75
6	0.000015	4.992850E-14	200	1784	00:31.91

Number of nodes = 40; Number of edges = 76.

Initial tolerance = 0.011388; Final tolerance = 0.000005 .

References

1. CHUNG, S. J., and MURTY, K. G., Polynomially Bounded Ellipsoid Algorithms for Convex Quadratic Programming, Nonlinear Programming 4, Edited by O. L. Mangasarian, R. R. Meyer, and S. M. Robinson, Academic Press, New York, New York, pp. 439-485, 1981.

2. DANTZIG, G. B., and COTTLE, R. W., Positive Semidefinite Programming, Nonlinear Programming, Edited by J. Abadie, North-Holland Publishing Company, Amsterdam, Holland, pp. 55-73, 1967.

3. LEMKE, C. E., On Complementary Pivot Theory, Mathematics of the Decision Sciences, Part 1, Edited by G. B. Dantzig and A. F. Veinott, American Mathematical Society, Providence, Rhode Island, pp. 95-114, 1968.

4. SHIAU, T. H., Iterative Linear Programming for Linear Complementarity and Related Problems, University of Wisconsin, Madison, Wisconsin, Computer Sciences Department, Ph.D. Thesis, 1983.

5. AHN, B. H., Solution of Nonsymmetric Linear Complementarity Problems by Iterative Methods I, Journal of Optimization Theory and Applications, Vol. 33, pp. 175-185, 1981.

6. CHENG, Y. C., Iterative Methods for Solving Linear Complementarity and Linear Programming Problems, University of Wisconsin, Madison, Wisconsin, Ph.D. Thesis, 1981.

7. COTTLE, R. W., GOLUB, G. H., and SACHER, R. S., On the Solution of Large Structured Complementarity Problems, Applied Mathematics and Optimization, Vol. 4, pp. 347-363, 1978.

8. CRYER, C. W., The Solution of a Quadratic Programming Problem Using Systematic Overrelaxation, SIAM Journal on Control, Vol. 9, pp. 385-392, 1971.

9. MANGASARIAN, O. L., Solution of Symmetric Linear Complementarity Problems by Iterative Methods, Journal of Optimization Theory and Applications, Vol. 22, pp. 465-485, 1977.

10. COTTLE, R. W., and DUVALL, S. G., A Lagrangian Relaxation Algorithm for the Constrained Matrix Problem, Stanford University, Stanford, California, Systems Optimization Laboratory Report 82-10, 1982.

11. HAN, S. P., and MANGASARIAN, O. L., A Dual Differentiable Exact Penalty Function, Mathematical Programming, Vol. 25, pp. 293-306, 1983.

12. PANG, J. S., On the Convergence of a Basic Iterative Method for the Implicit Complementarity Problem, Journal of Optimization Theory and Applications, Vol. 17, pp. 149-162, 1982.

13. FRANK, M., and WOLFE, P., An Algorithm for Quadratic Programming, Naval Research Logistics Quarterly, Vol. 3, pp. 95-110, 1956.

14. MANGASARIAN, O. L., and SHIAU, T. H., Error Bounds for Monotone Linear Complementarity Problems, Mathematical Programming, Vol. 36, pp. 81-89, 1986.

15. KARMARKAR, N., A New Polynomial-Time Algorithm for Linear Programming, Combinatorica, Vol. 4, pp. 373-395, 1984.

16. AGMON, S., The Relaxation Method for Linear Inequalities, Canadian Journal of Mathematics, Vol. 6, pp. 382-392, 1954.

17. MOTZKIN, T. S., and SCHOENBERG, I. J., The Relaxation Method for Linear Inequalities, Canadian Journal of Mathematics, Vol. 6, pp. 393-404, 1954.

18. CENSON, Y., and LENT, A., A Cyclic Subgradient Projections Method for the Convex Feasibility Problem, University of Haifa, Mt. Carmel, Haifa, Israel, Department of Mathematics Technical Report, July 1980.

19. ORTEGA, J. M., Numerical Analysis, A Second Course, Academic Press, New York, New York, 1972.

20. VARGA, R. S., Matrix Iterative Analysis, Prentice Hall, Incorporated, Englewood Cliffs, New Jersey, 1962.